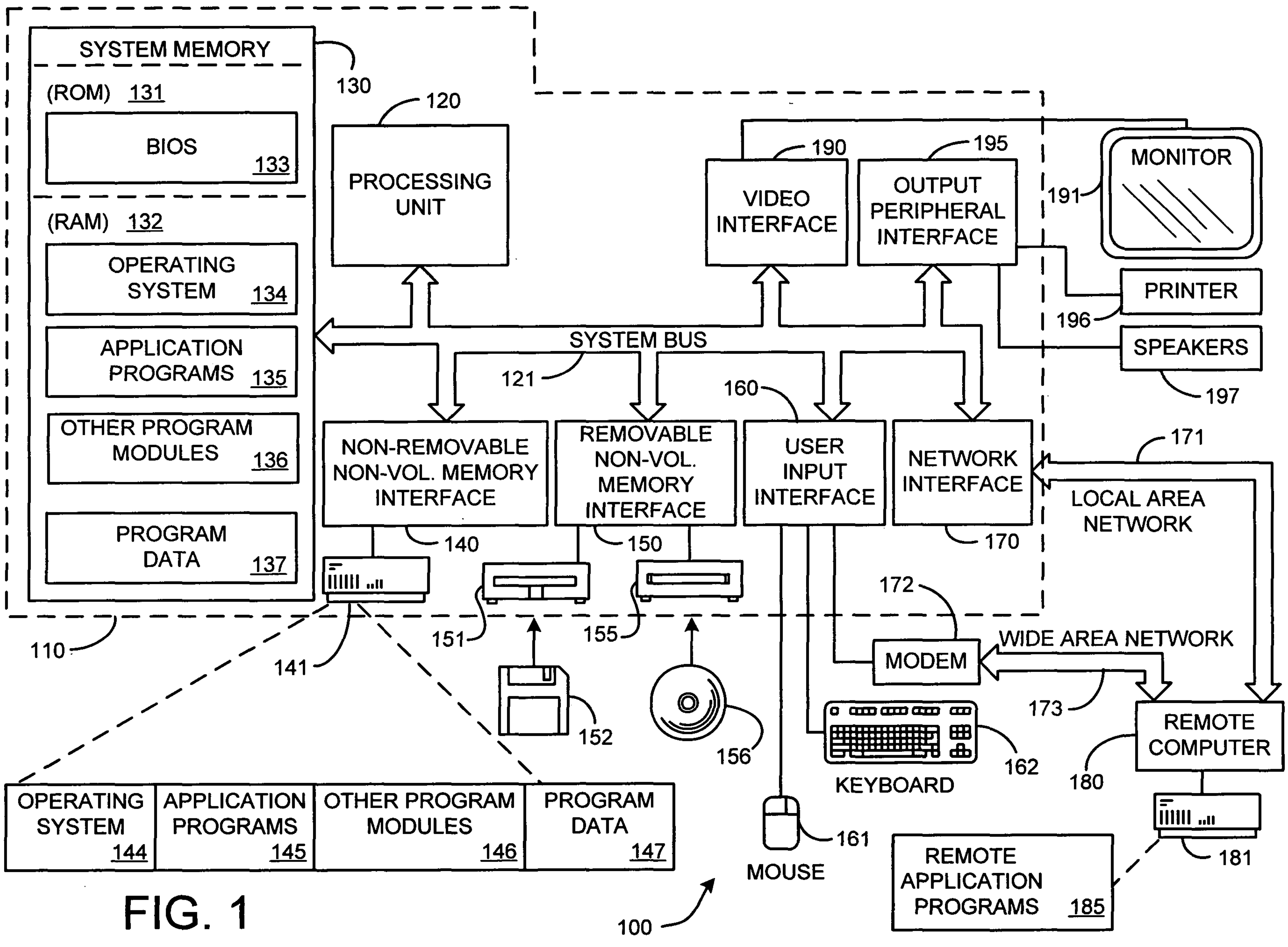

SYSTEM MEMORY
130
(ROM) 131
BIOS 133
(RAM) 132
OPERATING SYSTEM 134
APPLICATION PROGRAMS 135
OTHER PROGRAM MODULES 136
PROGRAM DATA 137
120
PROCESSING UNIT
190
VIDEO INTERFACE
195
OUTPUT PERIPHERAL INTERFACE
MONITOR
191
PRINTER
196
SPEAKERS
197
SYSTEM BUS
121
160
NON-REMOVABLE NON-VOL. MEMORY INTERFACE
REMOVABLE NON-VOL. MEMORY INTERFACE
USER INPUT INTERFACE
NETWORK INTERFACE
171
LOCAL AREA NETWORK
140
150
170
172
110
141
151
155
WIDE AREA NETWORK
MODEM
173
152
156
REMOTE COMPUTER
180
KEYBOARD
162
OPERATING SYSTEM 144
APPLICATION PROGRAMS 145
OTHER PROGRAM MODULES 146
PROGRAM DATA 147
MOUSE
161
181
REMOTE APPLICATION PROGRAMS 185
100

FIG. 1

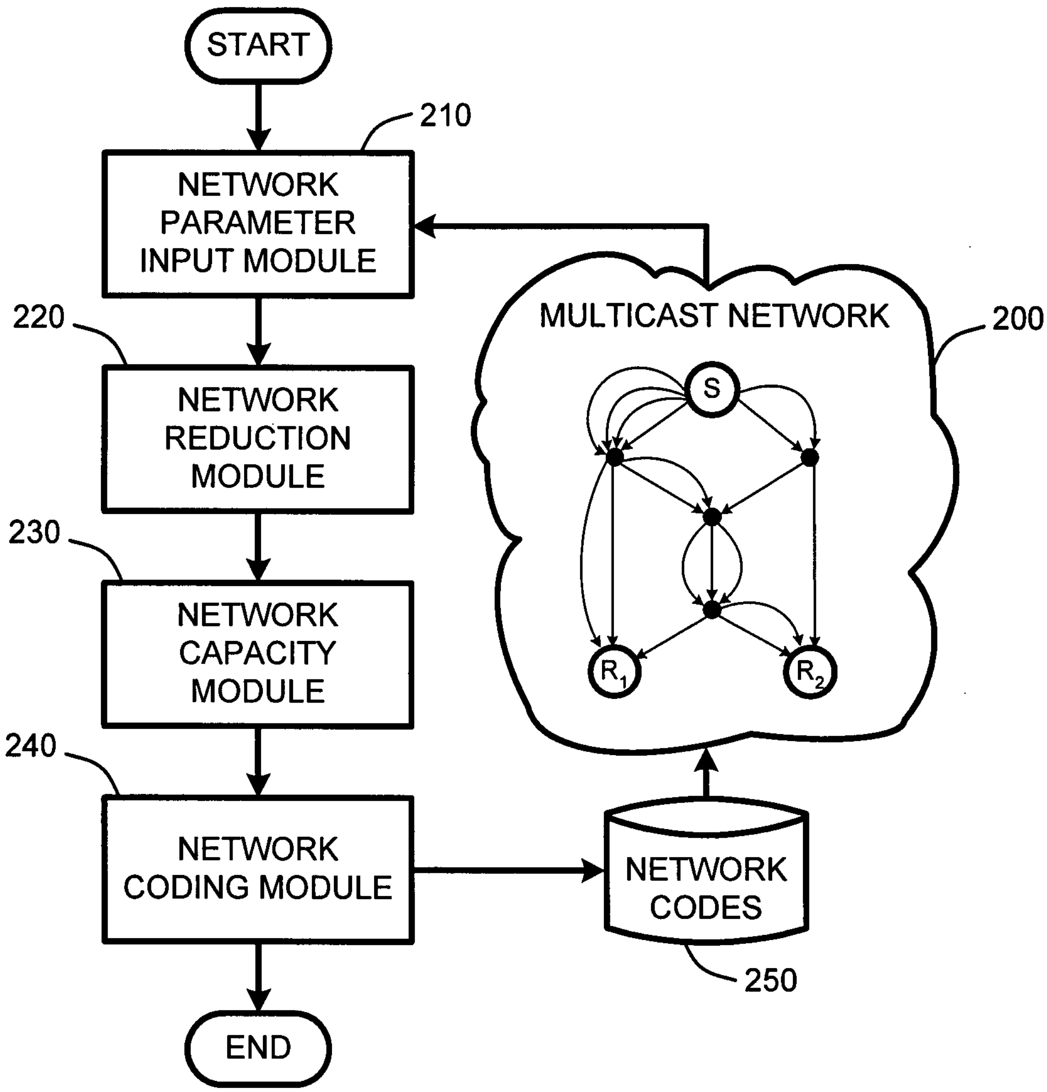

START
210
NETWORK PARAMETER INPUT MODULE
220
NETWORK REDUCTION MODULE
230
NETWORK CAPACITY MODULE
240
NETWORK CODING MODULE
END
MULTICAST NETWORK
200
S
R1
R2
NETWORK CODES
250

FIG. 2

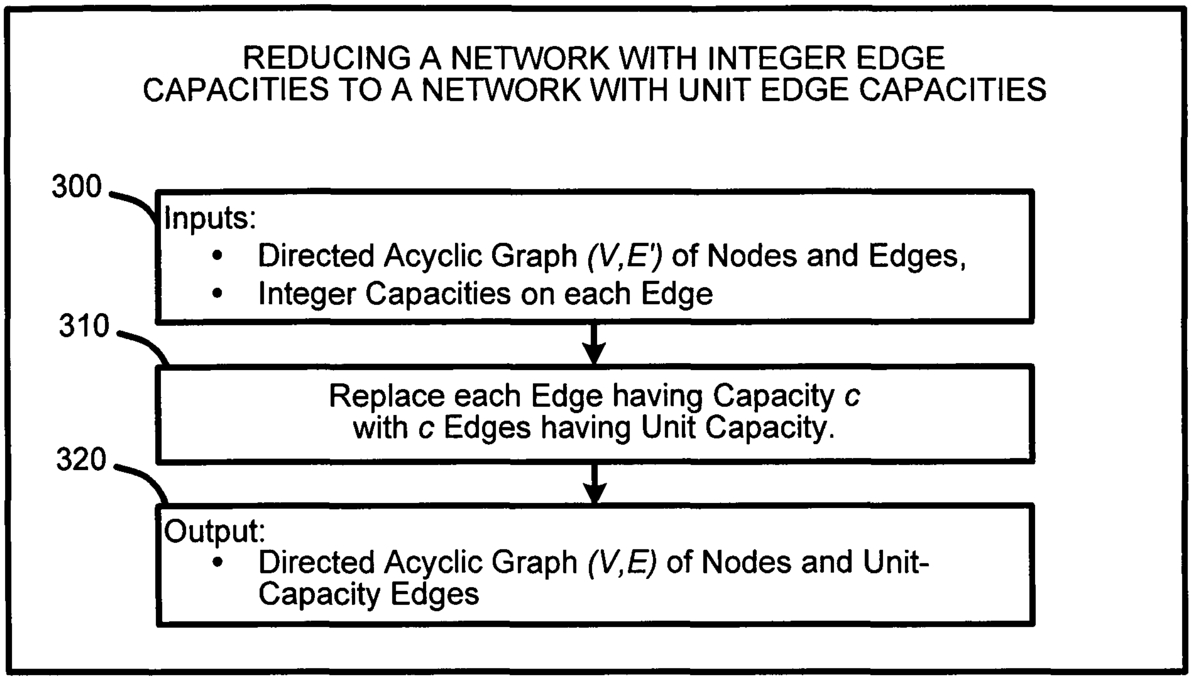

REDUCING A NETWORK WITH INTEGER EDGE CAPACITIES TO A NETWORK WITH UNIT EDGE CAPACITIES
300
Inputs:
• Directed Acyclic Graph (V,E') of Nodes and Edges,
• Integer Capacities on each Edge
310
Replace each Edge having Capacity c with c Edges having Unit Capacity.
320
Output:
• Directed Acyclic Graph (V,E) of Nodes and Unit-Capacity Edges

FIG. 3

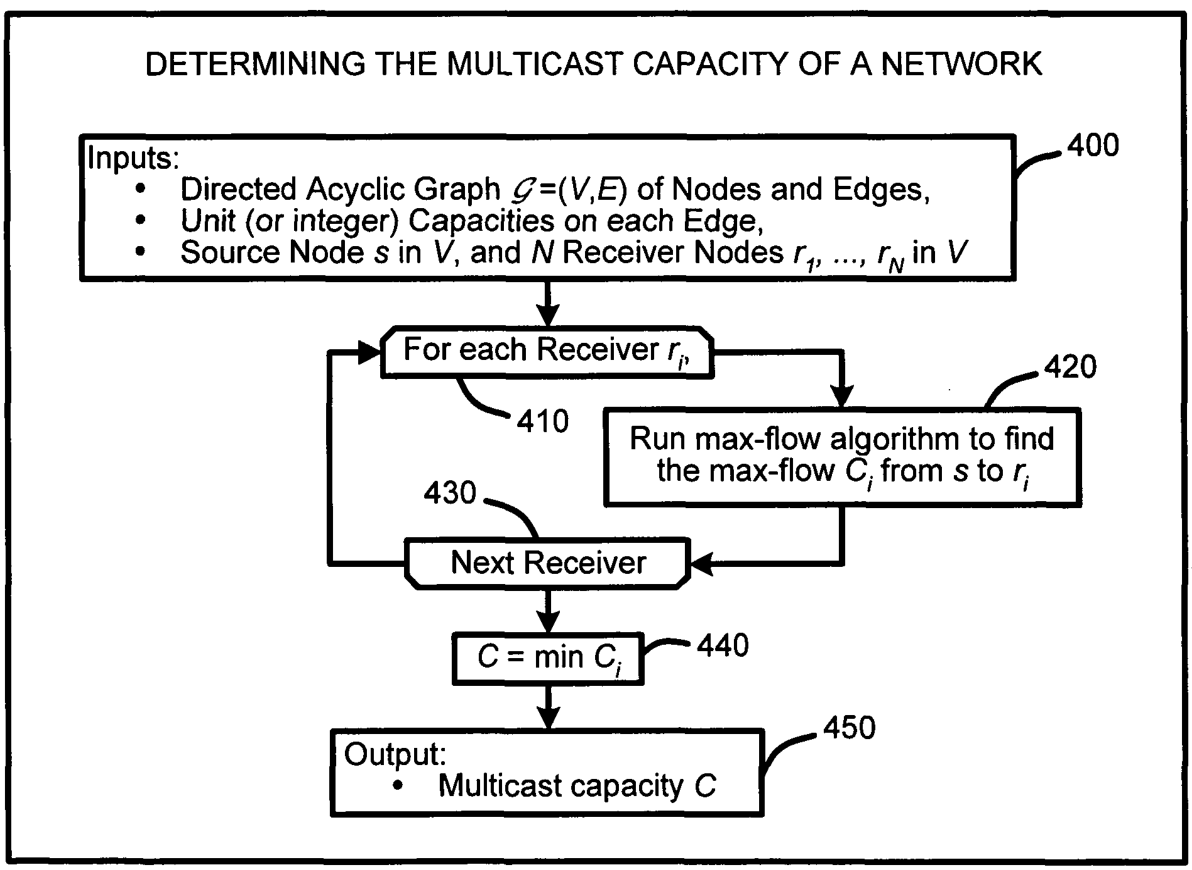
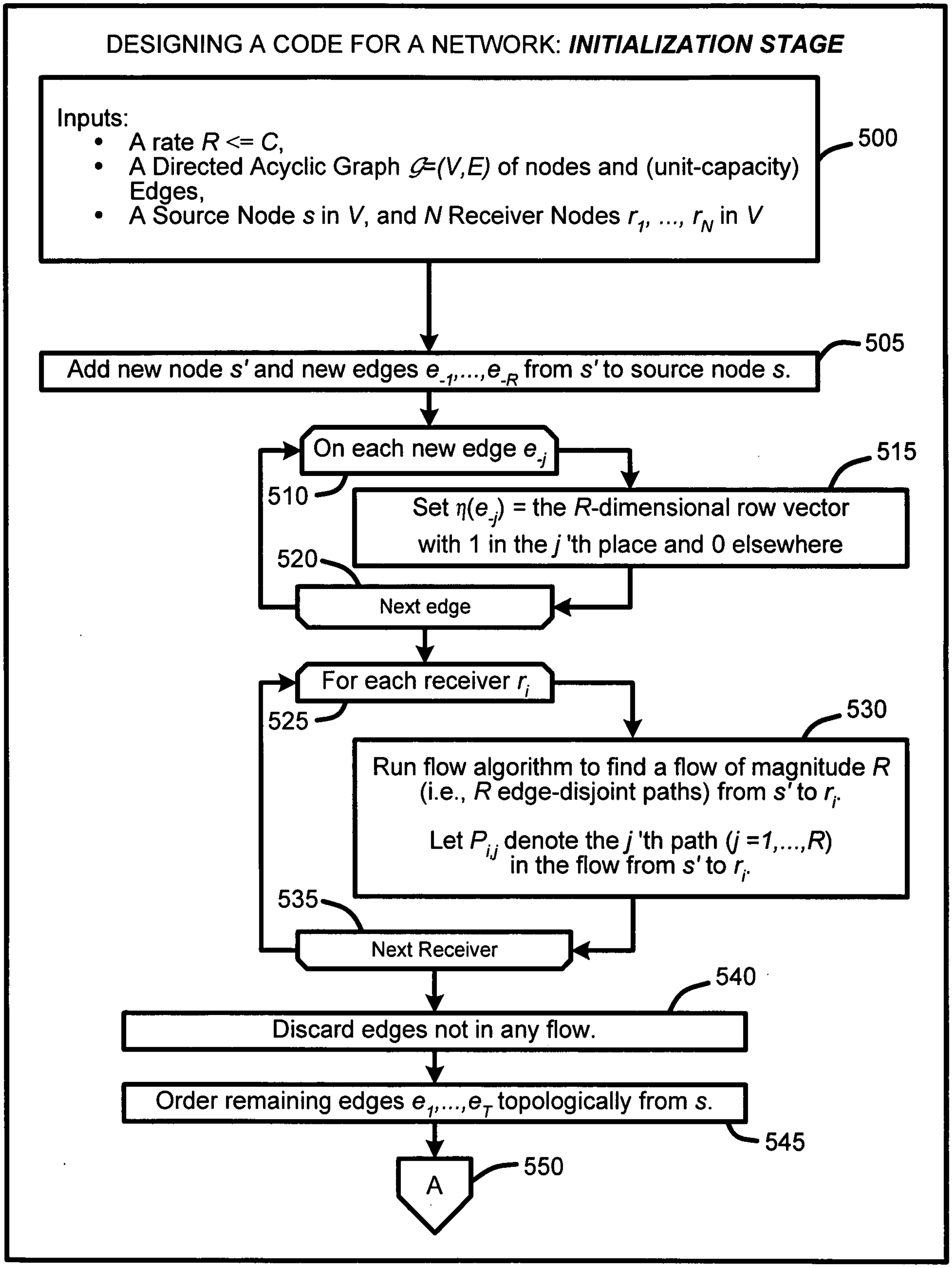
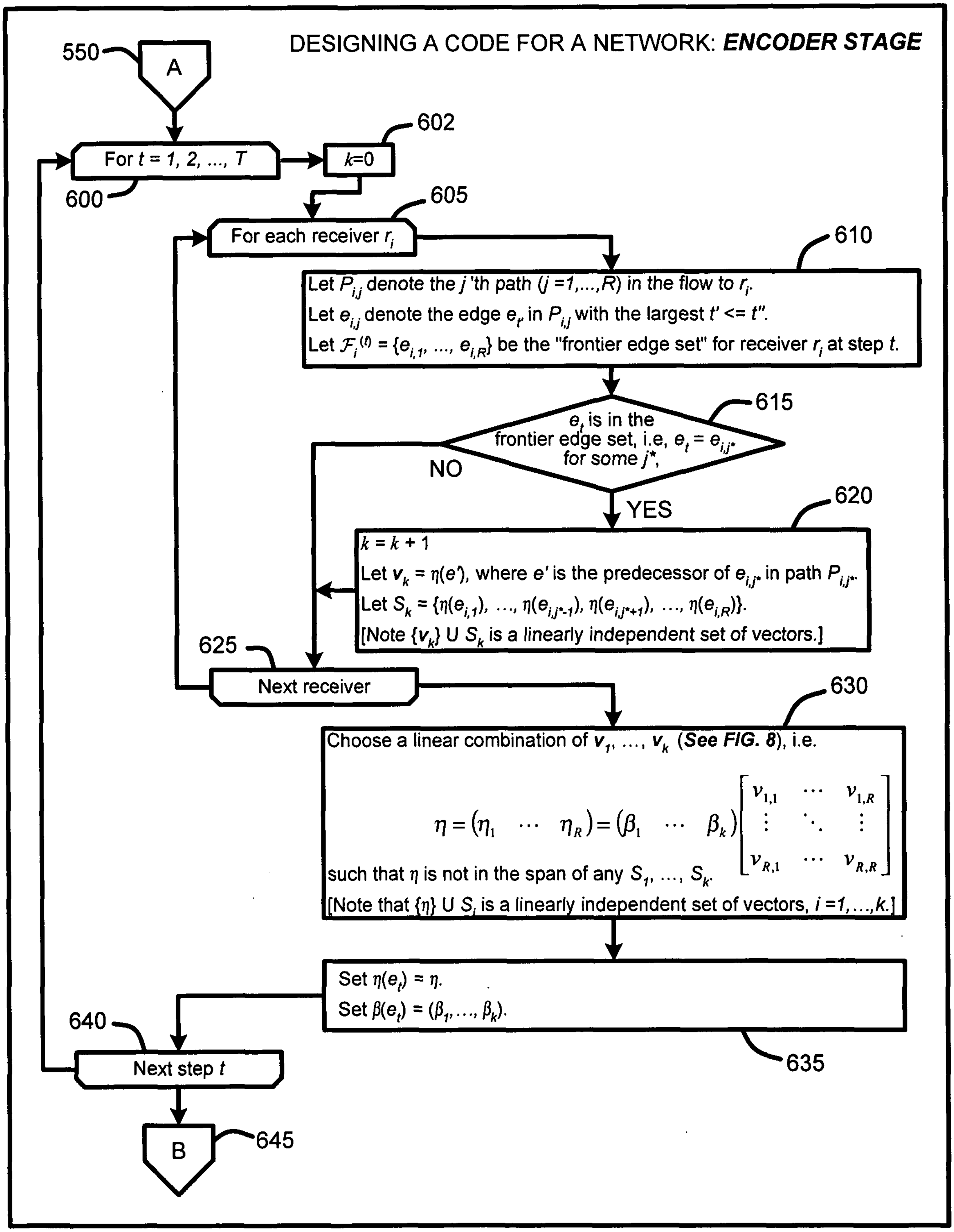

DETERMINING THE MULTICAST CAPACITY OF A NETWORK
Inputs:
• Directed Acyclic Graph $\mathcal{G}$=(V,E) of Nodes and Edges,
• Unit (or integer) Capacities on each Edge,
• Source Node s in V, and N Receiver Nodes r_1, ..., r_N in V
400
For each Receiver r_i,
410
420
Run max-flow algorithm to find the max-flow C_i from s to r_i
430
Next Receiver
C = min C_i
440
450
Output:
• Multicast capacity C

FIG. 4



DESIGNING A CODE FOR A NETWORK: INITIALIZATION STAGE
Inputs:
• A rate $R <= C$,
• A Directed Acyclic Graph $\mathcal{G}=(V,E)$ of nodes and (unit-capacity) Edges,
• A Source Node s in V, and N Receiver Nodes $r_1, ..., r_N$ in V
500
Add new node s' and new edges $e_{-1},...,e_{-R}$ from s' to source node s.
505
On each new edge e_{-j}
510
Set $\eta(e_{-j})$ = the R-dimensional row vector with 1 in the j 'th place and 0 elsewhere
515
520
Next edge
For each receiver r_i
525
Run flow algorithm to find a flow of magnitude R (i.e., R edge-disjoint paths) from s' to r_i.
Let $P_{i,j}$ denote the j 'th path ($j=1,...,R$) in the flow from s' to r_i.
530
535
Next Receiver
540
Discard edges not in any flow.
Order remaining edges $e_1,...,e_T$ topologically from s.
545
A
550

FIG. 5



DESIGNING A CODE FOR A NETWORK: ***ENCODER STAGE***
550
A
For t = 1, 2, ..., T
600
k=0
602
For each receiver r_i
605
610
Let $P_{i,j}$ denote the j'th path (j =1,...,R) in the flow to r_i.
Let $e_{i,j}$ denote the edge $e_{t'}$ in $P_{i,j}$ with the largest $t' <= t$".
Let $\mathcal{F}_i^{(t)} = \{e_{i,1}, ..., e_{i,R}\}$ be the "frontier edge set" for receiver r_i at step t.
615
e_t is in the frontier edge set, i.e, $e_t = e_{i,j^*}$ for some j^*,
NO
YES
620
$k = k + 1$
Let $\mathbf{v}_k = \eta(e')$, where e' is the predecessor of e_{i,j^*} in path P_{i,j^*}.
Let $S_k = \{\eta(e_{i,1}), ..., \eta(e_{i,j^*-1}), \eta(e_{i,j^*+1}), ..., \eta(e_{i,R})\}$.
[Note $\{\mathbf{v}_k\}$ U S_k is a linearly independent set of vectors.]
625
Next receiver
630
Choose a linear combination of $\mathbf{v}_1$, ..., $\mathbf{v}_k$ (***See FIG. 8***), i.e.
$$\eta = (\eta_1 \ \cdots \ \eta_R) = (\beta_1 \ \cdots \ \beta_k)\begin{bmatrix} v_{1,1} & \cdots & v_{1,R} \\ \vdots & \ddots & \vdots \\ v_{R,1} & \cdots & v_{R,R} \end{bmatrix}$$
such that η is not in the span of any S_1, ..., S_k.
[Note that $\{\eta\}$ U S_i is a linearly independent set of vectors, i =1,...,k.]
Set $\eta(e_t) = \eta$.
Set $\beta(e_t) = (\beta_1, ..., \beta_k)$.
635
640
Next step t
B
645

FIG. 6

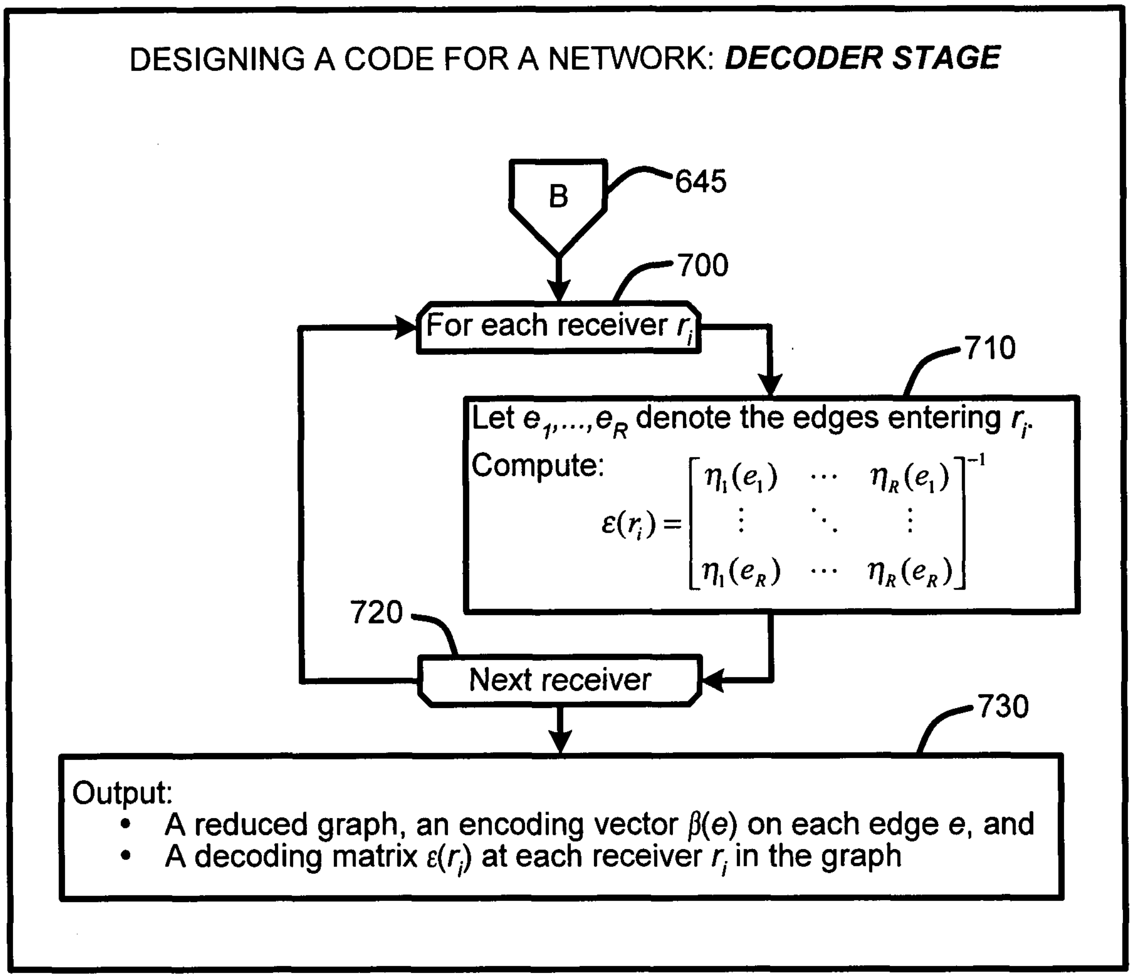

DESIGNING A CODE FOR A NETWORK: ***DECODER STAGE***
B
645
700
For each receiver r_i
710
Let $e_1,\ldots,e_R$ denote the edges entering r_i.
Compute:
$$\varepsilon(r_i) = \begin{bmatrix} \eta_1(e_1) & \cdots & \eta_R(e_1) \\ \vdots & \ddots & \vdots \\ \eta_1(e_R) & \cdots & \eta_R(e_R) \end{bmatrix}^{-1}$$
720
Next receiver
730
Output:
• A reduced graph, an encoding vector $\beta(e)$ on each edge e, and
• A decoding matrix $\varepsilon(r_i)$ at each receiver r_i in the graph

FIG. 7

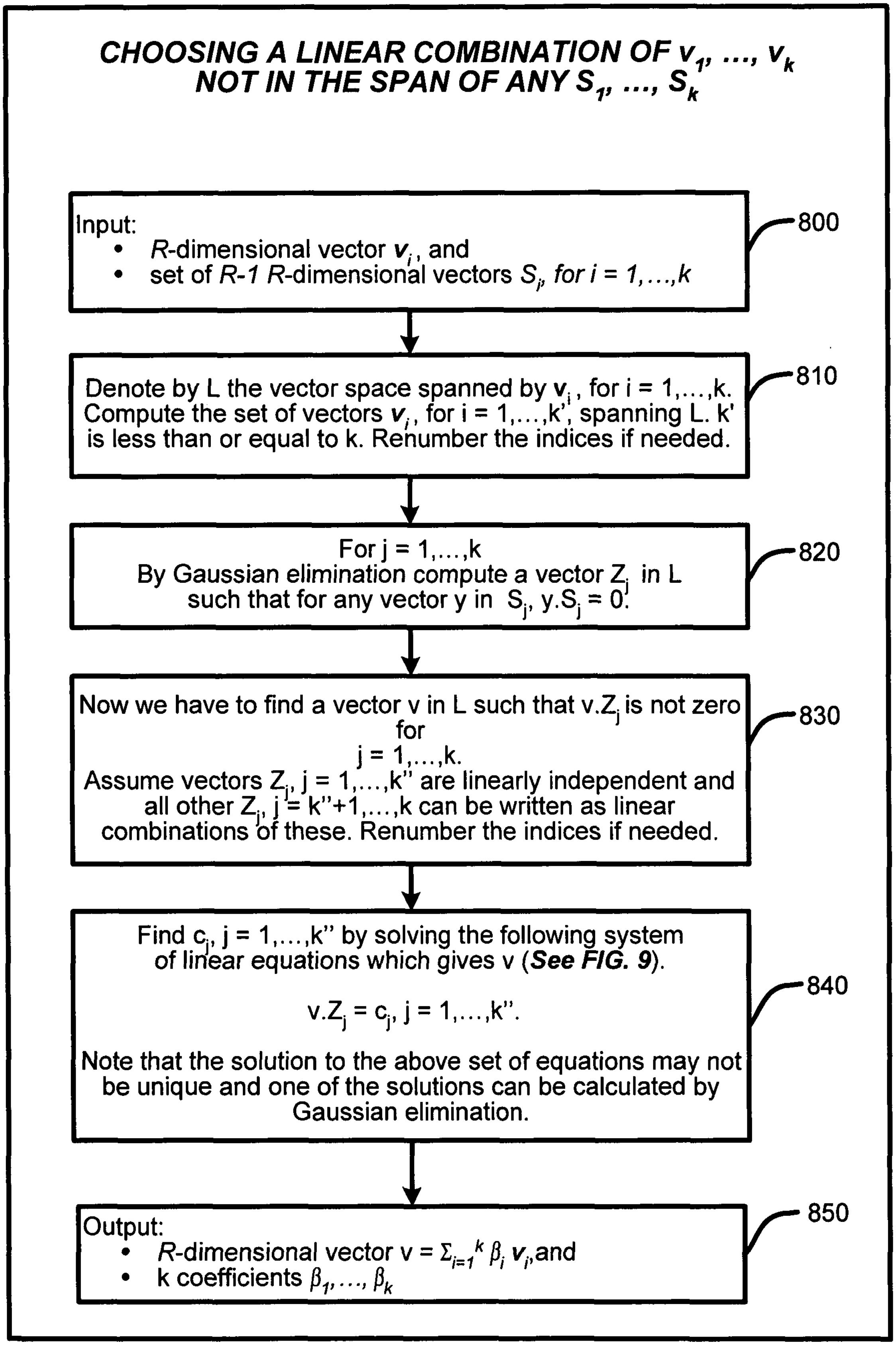

CHOOSING A LINEAR COMBINATION OF $v_1, \ldots, v_k$ NOT IN THE SPAN OF ANY $S_1, \ldots, S_k$
Input:
• R-dimensional vector v_i, and
• set of R-1 R-dimensional vectors S_i, for $i = 1,\ldots,k$
800
Denote by L the vector space spanned by v_i, for $i = 1,\ldots,k$. Compute the set of vectors v_i, for $i = 1,\ldots,k'$, spanning L. k' is less than or equal to k. Renumber the indices if needed.
810
For $j = 1,\ldots,k$
By Gaussian elimination compute a vector Z_j in L such that for any vector y in S_j, $y.S_j = 0$.
820
Now we have to find a vector v in L such that $v.Z_j$ is not zero for $j = 1,\ldots,k$.
Assume vectors Z_j, $j = 1,\ldots,k''$ are linearly independent and all other Z_j, $j = k''+1,\ldots,k$ can be written as linear combinations of these. Renumber the indices if needed.
830
Find c_j, $j = 1,\ldots,k''$ by solving the following system of linear equations which gives v (See FIG. 9).
$v.Z_j = c_j$, $j = 1,\ldots,k''$.
Note that the solution to the above set of equations may not be unique and one of the solutions can be calculated by Gaussian elimination.
840
Output:
• R-dimensional vector $v = \Sigma_{i=1}^{k} \beta_i v_i$, and
• k coefficients $\beta_1, \ldots, \beta_k$
850

FIG. 8

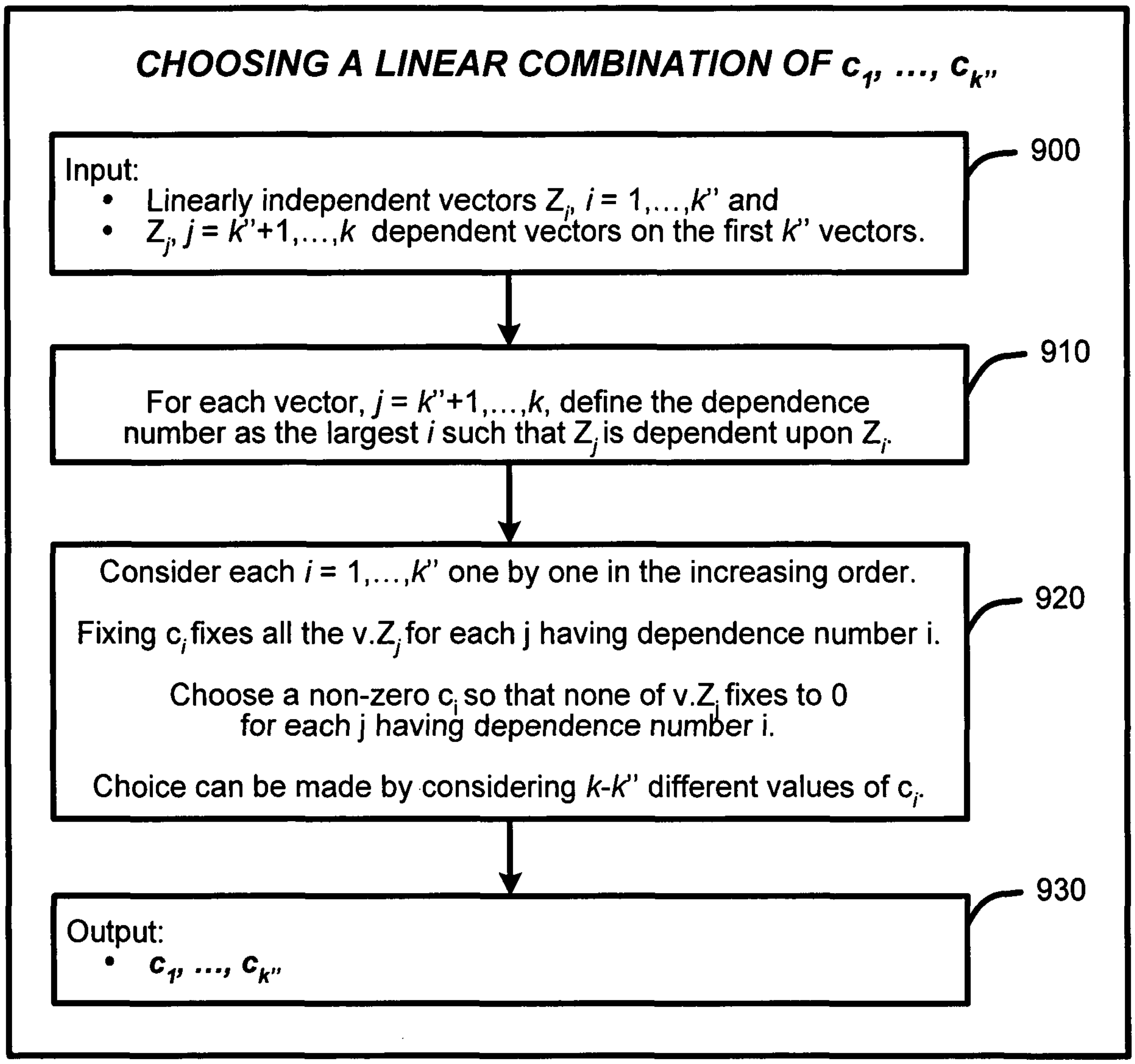

CHOOSING A LINEAR COMBINATION OF $c_1, \ldots, c_{k''}$
Input:
• Linearly independent vectors Z_i, $i = 1,\ldots,k'$ and
• Z_j, $j = k'+1,\ldots,k$ dependent vectors on the first k' vectors.
900
For each vector, $j = k'+1,\ldots,k$, define the dependence number as the largest i such that Z_j is dependent upon Z_i.
910
Consider each $i = 1,\ldots,k'$ one by one in the increasing order.
Fixing c_i fixes all the $v.Z_j$ for each j having dependence number i.
Choose a non-zero c_i so that none of $v.Z_j$ fixes to 0 for each j having dependence number i.
Choice can be made by considering k-k' different values of c_i.
920
Output:
• $c_1, \ldots, c_{k''}$
930

FIG. 9

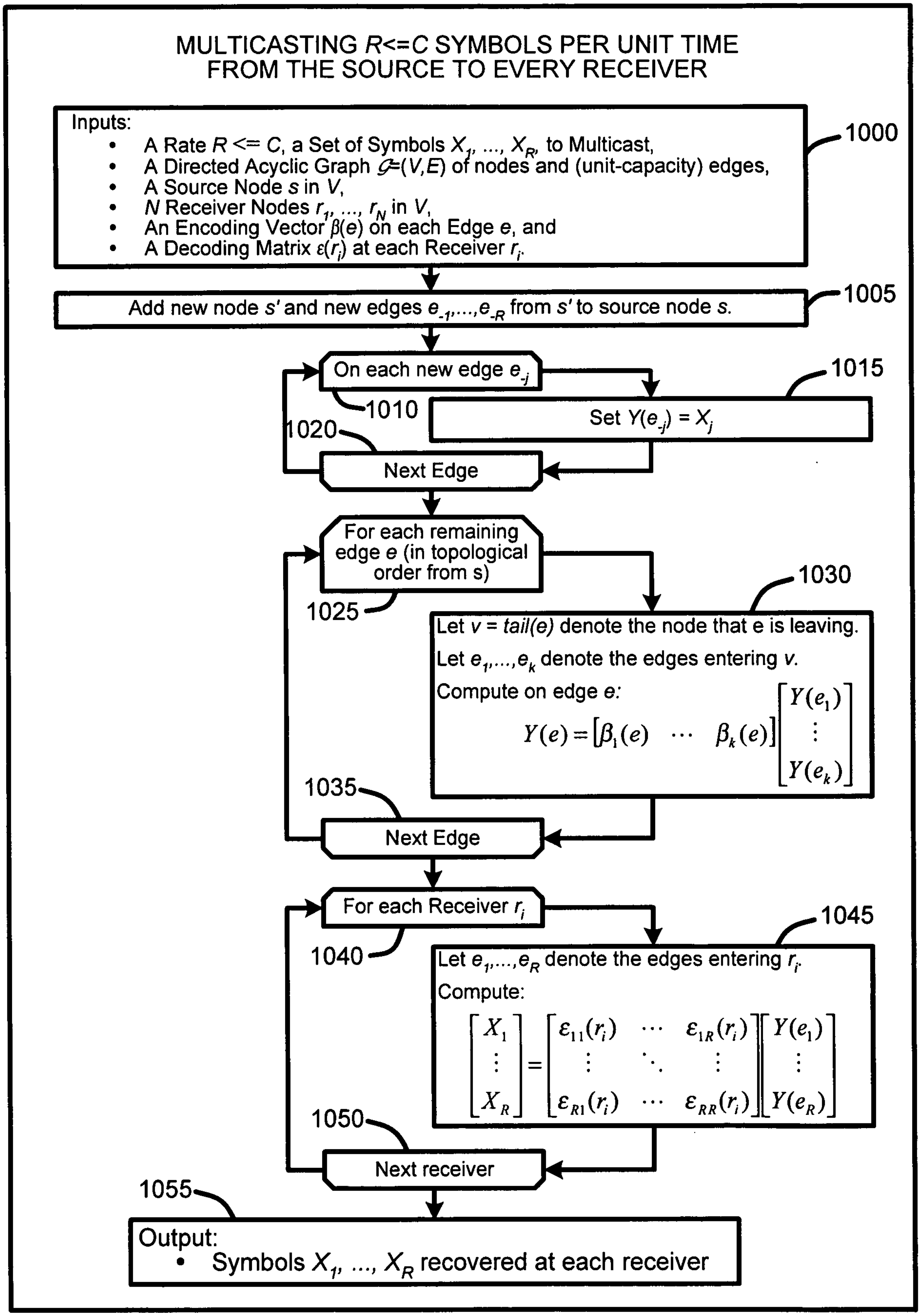

MULTICASTING R<=C SYMBOLS PER UNIT TIME FROM THE SOURCE TO EVERY RECEIVER
Inputs:
• A Rate R <= C, a Set of Symbols X_1, ..., X_R, to Multicast,
• A Directed Acyclic Graph $\mathcal{G}$=(V,E) of nodes and (unit-capacity) edges,
• A Source Node s in V,
• N Receiver Nodes r_1, ..., r_N in V,
• An Encoding Vector $\beta(e)$ on each Edge e, and
• A Decoding Matrix $\varepsilon(r_i)$ at each Receiver r_i.
1000
Add new node s' and new edges $e_{-1},\ldots,e_{-R}$ from s' to source node s.
1005
On each new edge e_{-j}
1010
Set $Y(e_{-j}) = X_j$
1015
1020
Next Edge
For each remaining edge e (in topological order from s)
1025
1030
Let v = $tail(e)$ denote the node that e is leaving.
Let $e_1,\ldots,e_k$ denote the edges entering v.
Compute on edge e:
$$Y(e)=\begin{bmatrix}\beta_1(e) & \cdots & \beta_k(e)\end{bmatrix}\begin{bmatrix}Y(e_1)\\ \vdots \\ Y(e_k)\end{bmatrix}$$
1035
Next Edge
For each Receiver r_i
1040
1045
Let $e_1,\ldots,e_R$ denote the edges entering r_i.
Compute:
$$\begin{bmatrix}X_1\\ \vdots \\ X_R\end{bmatrix}=\begin{bmatrix}\varepsilon_{11}(r_i) & \cdots & \varepsilon_{1R}(r_i)\\ \vdots & \ddots & \vdots \\ \varepsilon_{R1}(r_i) & \cdots & \varepsilon_{RR}(r_i)\end{bmatrix}\begin{bmatrix}Y(e_1)\\ \vdots \\ Y(e_R)\end{bmatrix}$$
1050
Next receiver
1055
Output:
• Symbols X_1, ..., X_R recovered at each receiver

FIG. 10